

FPT Approximation Schemes for Maximizing Submodular Functions

Piotr Skowron
University of Warsaw
Warsaw, Poland

Abstract

We investigate the existence of approximation algorithms for maximization of submodular functions, that run in fixed parameter tractable (FPT) time. Given a non-decreasing submodular set function $v : 2^X \rightarrow \mathbb{R}$ the goal is to select a subset S of K elements from X such that $v(S)$ is maximized. We identify three properties of set functions, referred to as p -separability properties, and we argue that many real-life problems can be expressed as maximization of submodular, p -separable functions, with low values of the parameter p . We present FPT approximation schemes for the minimization and maximization variants of the problem, for several parameters that depend on characteristics of the optimized set function, such as p and K . We confirm that our algorithms are applicable to a broad class of problems, in particular to problems from computational social choice, such as item selection or winner determination under several multiwinner election systems.

1 Introduction

We study (exponential-time) approximation algorithms for maximizing non-decreasing submodular set functions. A set function $v : 2^X \rightarrow \mathbb{R}$ is submodular if for each two subsets $A \subseteq B \subset X$ and each element $x \in X \setminus B$ it holds that $v(A \cup \{x\}) - v(A) \geq v(B \cup \{x\}) - v(B)$; v is non-decreasing if for each two subsets $A \subseteq B \subset X$ it holds that $v(A) \leq v(B)$. We consider the problem where the goal is to select a subset S of K elements from X such that the value $v(S)$ is maximal.

Maximization of non-decreasing submodular functions is a very general problem that is extensively used in various research areas, from recommendation systems [27, 35], through voting theory [36, 27], image engineering [17, 19, 32], information retrieval [41, 25], network design [21, 22], clustering [29], speech recognition [26], to sparse methods [3, 10]. Algorithms for maximization of non-decreasing submodular functions are applicable to other general problems of fundamental significance, such as the MAXCOVER problem [8, 34]. The universal relevance of the problem implies that the existence of good (approximation) algorithms for it is highly desired.

Indeed, the problem has already received a considerable amount of attention in the scientific community. For instance, it is known that the greedy algorithm, i.e., the algorithm that starts with the empty set and in each of K consecutive steps adds to the partial solution such an element from X that increases the value of the optimized function most, is an $(1 - 1/e)$ -approximation algorithm

for maximization of non-decreasing submodular functions [30]. The same approximation ratio can be achieved for the distributed [23] and online [37] variants of the problem. Algorithms for maximizing non-monotone submodular functions have been studied by Feige et al. [13], and the approximability of the problem with additional constraints has been investigated by Calinescu et al. [5], Sviridenko [38], Lee et al. [24], and Vondrák et al. [40]. Iwata et al. [15] have provided algorithmic view on minimizing submodular functions. For the survey on maximization of submodular functions we refer the reader to the work of Krause and Golovin [20].

Unfortunately, the approximation guarantees of the greedy algorithm cannot be improved without compromising the efficiency of computation. For example, the MAXCOVER problem can be expressed as maximization of a non-decreasing submodular function, yet it is known that under standard complexity assumptions no polynomial-time algorithm can approximate it better than with ratio $(1 - 1/e)$ [12]. Motivated by this fact, and provoked by the desire to obtain better approximation guarantees, we turn our attention to algorithms that run in super-polynomial time. In our studies we follow the approach of parameterized complexity theory and look for algorithms that run in fixed parameter tractable time (in FPT time), for some natural parameters. To the best of our knowledge, FPT approximation of optimizing submodular functions has not been considered in the literature before.

Parameterized complexity theory aims at investigating how the complexity of a problem depends on the size of different parts of input instances, called parameters. An algorithm runs in FPT time for a parameter P if it solves each instance I of the problem in time $O(f(|P|) \cdot \text{poly}(|I|))$, where f is a computable function. This definition excludes a large class of algorithms, such as the ones with complexity $O(|I|^{|P|})$. From the point of view of parameterized complexity, FPT is seen as the class of easy problems. Intuitively, the complexity of an FPT algorithm consists of two parts: $f(|P|)$, which is relatively low for small values of the parameter, and $\text{poly}(|I|)$ which is relatively low even for larger instances, because of polynomial relation between the computation time and the size of an instance. For details on parameterized complexity theory, we point the reader to appropriate overviews [11, 31, 14, 9].

We identify several parameters that we believe are suitable for a complexity analysis of maximization of non-decreasing submodular functions. Perhaps the most natural parameter to consider is the required size of solutions, K . Our other parameters depend on characteristics of the optimized set function. Specifically, we define a new property of set functions, called p -separability, and provide evidence that p is a natural parameter to consider. We do that in Section 4, by presenting several examples of real-life computational problems that can be expressed as maximization of submodular p -separable set functions, where the value of p is small.

Our main contribution is presentation and analysis of algorithms for the problem. We construct fixed parameter tractable approximation schemes, i.e., collections of algorithms that run in FPT time and that can achieve arbitrarily good approximation ratios. We provide algorithms for two variants of the problem: in the first variant, referred to as the *maximization variant*, the goal is to maximize the value $v(S)$. In the second one, referred to as the *minimization variant*, the goal is to minimize $(v(X) - v(S))$. While these two variants of the problem have the same optimal solutions, they are not equivalent in terms of their approximability. Indeed, if there exists a solution S with objectively high value, i.e., if $v(S)$ is close to $v(X)$, then approximation algorithm for the

minimization variant of the problem will be usually superior. For instance, if there exists a solution S such that $v(S) = 0.95 \cdot v(X)$, then a 2-approximation algorithm for the minimization variant of the problem is guaranteed to return a solution with the value better than $0.9 \cdot v(X)$. On the other hand, a $1/2$ -approximation algorithm for the maximization variant of the problem is allowed to return, in such a case, a solution with value $0.475 \cdot v(X)$. Conversely, if the value of an optimal solution is significantly lower than the value of the whole set X , then a good approximation algorithm for the maximization variant of the problem will produce solutions of a better quality.

Our algorithms run in FPT time for the parameter (K, p) , where K is the size of the solution, and p is the lowest value such that the set function is p -separable. To address the case of functions which are not p -separable for any reasonable values p , we define a weaker form of approximability, referred to as approximation of the *minimization-or-maximization variant*—here, the goal is to find a subset S that is good in one of the previous two metrics. Such algorithms are also desired as they are guaranteed to find good approximation solutions, provided high quality solutions exist (i.e., if values of the optimal solutions are close to $v(X)$). We show that there exists a randomized FPT approximation scheme for minimization-or-maximization variant of the problem for the parameter $(K, \sum_{x \in X} v(\{x\})/v(X))$.

We believe that the consequences of our general results are quite significant. In particular, in Section 4, we prove the existence of FPT approximation schemes for some natural problems in the computational social choice, in the matching theory, and in the theoretical computer science.

2 Notation and Definitions

Let X denote the universe set. We consider a set function $v : 2^X \rightarrow \mathbb{R}$ that is non-negative, i.e., such that for each $S \subseteq X$ we have $v(S) \geq 0$. We say that a function v is non-decreasing if for each two subsets $A \subseteq B \subseteq X$ it holds that $v(B) \geq v(A)$. A set function v is submodular if for each two subsets $A \subseteq B \subset X$ and each element $x \in X \setminus B$ it holds that $v(A \cup \{x\}) - v(A) \geq v(B \cup \{x\}) - v(B)$. There are numerous equivalent conditions characterizing submodular functions—for a survey we refer the reader to the seminal article of Nemhauser et. al. [30]. It is easy to see that if the set function v is non-decreasing and submodular, then for each two subsets $A \subseteq B \subset X$ and each element $x \in X$ it holds that $v(A \cup \{x\}) - v(A) \geq v(B \cup \{x\}) - v(B)$ (here, we do not have to assume that $x \in X \setminus B$).

Below, we define a new class of properties of set functions that we call p -separability.

Definition 2.1 (p -separable set function). *A submodular set function $v : 2^X \rightarrow \mathbb{R}$ is:*

1. p -superseparable, if for each $S \subseteq X$ we have:

$$\sum_{x \in X} (v(S \cup \{x\}) - v(S)) \geq \sum_{x \in X} v(\{x\}) - p \cdot v(S), \quad (1)$$

2. at-least- p -subseparable, if for each $S \subseteq X$ we have:

$$\sum_{x \in X} (v(S \cup \{x\}) - v(S)) \geq p \cdot v(X) - p \cdot v(S). \quad (2)$$

3. at-most- p -subseparable, if for each $S \subseteq X$ we have:

$$\sum_{x \in X} (v(S \cup \{x\}) - v(S)) \leq p \cdot v(X) - p \cdot v(S). \quad (3)$$

If we think of the universe set X as of a space of actions, and of the set function v as of a function measuring the joint effects of sets of actions, then submodularity has a natural intuitive interpretation: it says that the marginal contribution of performing each action decreases as the set of actions already performed grows. We can also provide intuitive interpretation of p -separability. A set function v is p -superseparable if for each state S , representing the set of already performed actions, the sum of effects of all the actions performed in S is lower bounded by the sum of marginal contributions of all the actions performed individually, minus the contribution of actions from S performed together multiplied by p . A set function v is *at-least- p -subseparable* (*at-most- p -subseparable*) if the the sum of marginal contributions of all the actions performed in each state S is lower-bounded (upper-bounded) by p multiplied by the contribution of these actions if performed together, $v(X) - v(S)$. In Section 4 we argue that the assumption about p -separability of set functions is plausible: we give examples of several algorithmic problems that can be formulated as optimization tasks for p -separable submodular functions.

We observe that a linear combination of p -superseparable functions is p -superseparable. The same comment applies to at-least- p -subseparability and at-most- p -subseparability. As we will see in Section 4, this observation is helpful in proving that certain set functions are p -separable.

In this paper we investigate the problem of selecting K elements from X that, altogether, maximize the value of the set function v .

Definition 2.2 (BESTKSUBSET). *For a set of elements X , a polynomially computable set function $v : 2^X \rightarrow \mathbb{R}$, and an integer K , the solution to the BESTKSUBSET problem is such a set $S \subseteq X$ that $\|S\| \leq K$ and that $v(S)$ is maximal.*

We are specifically interested in finding approximation algorithms for the BESTKSUBSET problem. We focus on approximating two metrics: (i) the value $v(S)$ in the maximization variant of the problem, and (ii) the value $(v(X) - v(S))$ in its minimization variant.

Definition 2.3 (Approximation algorithms). *Let S^* denote an optimal solution for BESTKSUBSET:*

1. Fix α , $0 < \alpha < 1$. \mathcal{A} is an α -approximation algorithm for the maximization variant of BESTKSUBSET, if for each instant I of BESTKSUBSET it returns a set S such that $v(S) \geq \alpha v(S^*)$.
2. Fix α , $\alpha > 1$. \mathcal{A} is an α -approximation algorithm for the minimization variant of BESTKSUBSET, if for each instant I of BESTKSUBSET it returns a set S such that $(v(X) - v(S)) \leq \alpha(v(X) - v(S^*))$.
3. Fix α , $\alpha > 1$. \mathcal{A} is an α -approximation algorithm for the minimization-or-maximization variant of BESTKSUBSET, if for each instant I of BESTKSUBSET it returns a set S such that $v(S) \geq \frac{1}{\alpha} v(S^*)$ or $(v(X) - v(S)) \leq \alpha(v(X) - v(S^*))$.

The definition of an approximation algorithm for minimization-or-maximization variant of BESTKSUBSET requires some additional comment: this definition guarantees that the algorithm finds a good solution provided a high quality solution exists. In other words, if there exists an optimal solution S^* such that the value $(v(X) - v(S^*))$ is low compared to $v(S^*)$, then the good approximation solution for the minimization variant of the problem is also a good solution for its maximization variant. For some parameters we present good approximation algorithms for the minimization-or-maximization variant of BESTKSUBSET, even though we do not have as good algorithms neither for the minimization nor maximization variants of the problem.

We are specifically interested in FPT approximation schemes. A collection of algorithms \mathcal{A} is an FPT approximation scheme for a parameter P , if for each constant α there exists an α -approximation algorithm in \mathcal{A} that runs in an FPT time for the parameter P .

3 Algorithms for Maximizing p -separable Submodular Functions

In this section we present our approximation algorithms for the three variants of the problem, formally stated in Definition 2.3, of the BESTKSUBSET problem. Our methods are inspired by the algorithms of Skowron and Faliszewski [34] for the MAXCOVER problem. We extend these algorithms to be applicable to the problem of maximizing more general submodular functions.

We start with presenting an FPT approximation scheme for BESTKSUBSET for submodular p -superseparable set functions. The algorithm, formally defined as Algorithm 1, gets as an input an instance of the problem and the required approximation ratio, β . It proceeds in two steps: first, it restricts the universe set by selecting a certain number of elements from X with the highest values of the set function v . Second, it takes the set \mathcal{A} of elements that were selected in the first step, computes the value of the set function for all K -element subsets of \mathcal{A} , and returns a subset with the highest value.

Algorithm 1 is an FPT approximation scheme for the maximization variant of the problem for the parameter (K, p) . Before we prove this fact, however, we note that under standard complexity theoretic assumptions, there exists no FPT exact algorithm for the problem. There even exists no FPT exact algorithm for the parameter K if p is a constant. This follows from our observation in Section 4.3, where we show that the MAXCOVER problem with frequencies bounded by p can be expressed as maximization of a non-negative, non-decreasing, submodular, p -superseparable set function, and from the fact that the MAXCOVER problem with frequencies bounded by a constant, for the parameter K belongs to the complexity class $W[1]$ [34], and it is unlikely that $W[1] \subseteq \text{FPT}$.

Theorem 3.1. *For each non-negative, non-decreasing, submodular, and p -superseparable set function $v : 2^X \rightarrow \mathbb{R}$ and for each $0 \leq \beta < 1$, Algorithm 1 outputs a β -approximate solution for the maximization variant of BESTKSUBSET, in time $\text{poly}(n, m) \cdot \binom{pK}{(1-\beta)K} + K$.*

Proof. Consider an input instance I of the BESTKSUBSET problem. Let S and S^* be, respectively, the solution returned by Algorithm 1 and some optimal solution. We set $\text{OPT} = v(S^*)$ as the value of an optimal solution.

Algorithm 1: An algorithm for the BESTKSUBSET problem for non-negative, non-decreasing, submodular, and p -superseparable set functions.

Parameters:

X — the set of elements.

v — the submodular function $v : 2^X \rightarrow \mathbb{R}$ that is p -superseparable.

β — the required approximation ratio of the algorithm.

$\mathcal{A} \leftarrow \lceil \frac{pK}{(1-\beta)} + K \rceil$ elements x from X with highest values $v(\{x\})$;

return K -element subset of \mathcal{A} with the highest value of v ;

We will show that $v(S) \geq \beta \text{OPT}$. Naturally, the value $v(S)$ might be lower than $v(S^*)$. This might happen because \mathcal{A} , the set of the elements considered by the algorithm in its second step, might not contain some elements from S^* . We will show that $\ell = |S^* \setminus \mathcal{A}|$ elements from $S^* \setminus \mathcal{A}$ might be replaced by some elements from \mathcal{A} which are not present in S^* , in a way that decreases the value of S^* by at most a small fraction. After such replacement, we will end up with the set containing the elements from \mathcal{A} only. From this we will infer that the value of the best solution in \mathcal{A} is lower than the value of an optimal solution by at most a small factor.

Let us order the elements from $S^* \setminus \mathcal{A}$ in some arbitrary way, and let us use the notation $S^* \setminus \mathcal{A} = \{x_1, \dots, x_\ell\}$. We will replace the elements $\{x_1, \dots, x_\ell\}$ with the elements $\{x'_1, \dots, x'_\ell\}$ (we will define these elements later), one by one, in ℓ consecutive steps. Thus, in the i -th step we will replace x_i with x'_i in the set $(S^* \setminus \{x_1, \dots, x_{i-1}\}) \cup \{x'_1, \dots, x'_{i-1}\}$. The elements x'_1, \dots, x'_ℓ are defined by induction, in the following way. Assume that we have already found elements x'_1, \dots, x'_{i-1} (for $i = 1$ it means we have not yet found any element, i.e., that we are looking for the first element in the sequence). We define x'_i to be an element from $\mathcal{A} \setminus (S^* \cup \{x'_1, \dots, x'_{i-1}\})$ that maximizes the value $v((S^* \setminus \{x_1, \dots, x_i\}) \cup \{x'_1, \dots, x'_i\})$.

It may happen that after replacing x_i with x'_i , the value of the function v for the new set decreases. Let Δ_i denote the value of such decrease (or increase if the algorithm were lucky—in such case Δ_i would be negative):

$$\begin{aligned} \Delta_i = & v((C^* \setminus \{x_1, \dots, x_{i-1}\}) \cup \{x'_1, \dots, x'_{i-1}\}) \\ & - v((C^* \setminus \{x_1, \dots, x_i\}) \cup \{x'_1, \dots, x'_i\}). \end{aligned}$$

By the construction of the set \mathcal{A} and the fact that $x_i \notin \mathcal{A}$, for every $y \in \mathcal{A} \setminus (S^* \cup \{x'_1, \dots, x'_{i-1}\})$ we have that $v(\{x_i\}) \leq v(\{y\})$. By the way we choose the element x'_i , we know that for every $y \in \mathcal{A} \setminus (S^* \cup \{x'_1, \dots, x'_{i-1}\})$, we have:

$$\begin{aligned} \Delta_i \leq & v((C^* \setminus \{x_1, \dots, x_{i-1}\}) \cup \{x'_1, \dots, x'_{i-1}\}) \\ & - v((C^* \setminus \{x_1, \dots, x_i\}) \cup \{x'_1, \dots, x'_{i-1}, y\}). \end{aligned}$$

Using submodularity and after reformulation we get:

$$\Delta_i \leq v((C^* \setminus \{x_1, \dots, x_i\}) \cup \{x'_1, \dots, x'_{i-1}\}) + v(\{x_i\}) - v(\emptyset)$$

$$\begin{aligned}
& -v\left((\mathcal{C}^* \setminus \{x_1, \dots, x_i\}) \cup \{x'_1, \dots, x'_{i-1}, y\}\right) \\
& \leq v\left((\mathcal{C}^* \setminus \{x_1, \dots, x_i\}) \cup \{x'_1, \dots, x'_{i-1}\}\right) + v(\{y\}) - v(\emptyset) \\
& \quad - v\left((\mathcal{C}^* \setminus \{x_1, \dots, x_i\}) \cup \{x'_1, \dots, x'_{i-1}, y\}\right).
\end{aligned}$$

For $y \notin X$, by submodularity and monotonicity, we have that:

$$\begin{aligned}
0 & \leq v\left((\mathcal{C}^* \setminus \{x_1, \dots, x_i\}) \cup \{x'_1, \dots, x'_{i-1}\}\right) + v(\{y\}) - v(\emptyset) \\
& \quad - v\left((\mathcal{C}^* \setminus \{x_1, \dots, x_i\}) \cup \{x'_1, \dots, x'_{i-1}, y\}\right).
\end{aligned}$$

Since the set function is non-negative, the inequalities above will still hold if we skip the fragment $v(\emptyset)$. Consequently, since the set function is p -superseparable, we get:

$$\begin{aligned}
(\|\mathcal{A}\| - K)\Delta_i & \leq \sum_y \left(v\left((\mathcal{C}^* \setminus \{x_1, \dots, x_i\}) \cup \{x'_1, \dots, x'_{i-1}\}\right) + v(\{y\}) \right. \\
& \quad \left. - v\left((\mathcal{C}^* \setminus \{x_1, \dots, x_i\}) \cup \{x'_1, \dots, x'_{i-1}, y\}\right) \right) \\
& \leq p \cdot v\left((\mathcal{C}^* \setminus \{x_1, \dots, x_i\}) \cup \{x'_1, \dots, x'_{i-1}\}\right) \leq p\text{OPT}.
\end{aligned}$$

Which leads to:

$$\Delta_i \leq \frac{p\text{OPT}}{\|\mathcal{A}\| - K} = \frac{\text{OPT}p(1 - \beta)}{pK} = \frac{\text{OPT}(1 - \beta)}{K}.$$

Since $\ell \leq K$, we conclude that:

$$\sum_{i=1}^{\ell} \Delta_i \leq (1 - \beta)\text{OPT}.$$

That is, after replacing the elements from S^* that do not appear in \mathcal{A} with sets from \mathcal{A} , the optimal value is decreased by at most $(1 - \beta)\text{OPT}$. This means that there are K elements in \mathcal{A} for which the function v achieves the value equal to at least βOPT . Since the algorithm tries all size- K subsets of \mathcal{A} , it finds a solution with such a value. \square

Now, we move to optimizing at-most- p -subseparable set functions. We know that the simple greedy algorithm (Algorithm 2) achieves approximation ratio of $(1 - e^{-1})$ for the maximization variant of the BESTKSUBSET problem [30]. Below, we show that in some cases the analysis of this approximation guarantees can be improved for the case of at-most- p -subseparable set functions. We note that in this case we do not even require the set function to be submodular.

Theorem 3.2. *The greedy algorithm (Algorithm 2) is a polynomial-time $(1 - e^{-\frac{pK}{|\mathcal{X}|}})$ -approximation algorithm for the maximization variant of BESTKSUBSET problem with non-negative, non-decreasing, at-least- p -subseparable set function.*

Algorithm 2: An algorithm for the BESTKSUBSET problem for non-negative, non-decreasing, at-least- p -subseparable set functions.

Parameters:

X — the set of elements.

v — the submodular function $v : 2^X \rightarrow \mathbb{R}$ that is at-least- p -subseparable.

$C = \{\};$

for $i \leftarrow 1$ **to** K **do**

$C \leftarrow C \cup \left\{ \operatorname{argmax}_{x \in X} \left(v(C \cup \{x\}) - v(C) \right) \right\}$

return C

Proof. The algorithm clearly runs in polynomial time and so we focus only on proving its approximation ratio.

We prove by induction that for each i , $0 \leq i \leq K$, after the i 'th iteration of the greedy algorithm's "for" loop, the value $(v(X) - v(C))$ is at most equal to $(v(X) - v(\emptyset)) \left(1 - \frac{p}{|X|}\right)^i$. Naturally, the assumption is true for $i = 0$. Suppose that the inductive assumption holds for some $(i - 1)$, $1 \leq i < K$. Let S be the partial solution at the beginning of the i -th iteration of the algorithm's "for" loop and let x_b be the element added to the partial solution in this iteration. Since the set function v is at-least- p -subseparable, it holds that:

$$\sum_{x \in X} (v(S \cup \{x\}) - v(S)) \geq p \cdot v(X) - p \cdot v(S).$$

From the pigeonhole principle we get that:

$$(v(S \cup \{x_b\}) - v(S)) \geq \frac{p}{|X|} (v(X) - v(S)).$$

Thus, we get that:

$$\begin{aligned} v(X) - v(S \cup \{x_b\}) &\leq v(X) - v(S) + v(S) - v(S \cup \{x_b\}) \\ &\leq v(X) - v(S) - \frac{p}{|X|} (v(X) - v(S)) \\ &= (v(X) - v(S)) \left(1 - \frac{p}{|X|}\right) \leq (v(X) - v(\emptyset)) \left(1 - \frac{p}{|X|}\right)^i. \end{aligned}$$

Let C be the solution returned by Algorithm 2.

$$v(X) - v(C) \leq (v(X) - v(\emptyset)) \left(1 - \frac{p}{|X|}\right)^{\frac{|X|}{p} \cdot \frac{pK}{|X|}} \leq (v(X) - v(\emptyset)) e^{-\frac{pK}{|X|}}.$$

Since the function v is monotonic, and thus $\text{OPT} \leq v(X)$, and since v is non-negative, and thus $v(\emptyset) \geq 0$, we have that:

$$v(C) \geq v(X) - (v(X) - v(\emptyset)) e^{-\frac{pK}{|X|}} \geq \text{OPT} \left(1 - e^{-\frac{pK}{|X|}}\right).$$

This completes the proof. □

Algorithm 3: An algorithm for the minimization variant of the BESTKSUBSET problem with a non-negative, non-decreasing, submodular, and at-most- p -subseparable set function.

Parameters:

X — the set of elements.

v — the submodular function $v : 2^X \rightarrow \mathbb{R}$ that is at-most- p -subseparable.

β — the required approximation ratio of the algorithm

ϵ — the allowed probability of achieving worse than β approximation ratio

SingleRun():

$S \leftarrow \emptyset$;

for $i \leftarrow 0$ **to** K **do**

$x_r \leftarrow$ randomly select an element from $X \setminus S$

 with probability of selecting x proportional to $v(S \cup \{x\}) - v(S)$;

$S \leftarrow S \cup \{x_r\}$;

return S ;

Main() : run SingleRun() for $\lceil -\ln \epsilon / (\frac{\beta-1}{p\beta})^K \rceil$ times; return the best solution;

If we additionally assume that the set function is submodular, then naturally, the standard approximation ratio of $(1 - e^{-1})$ of the greedy algorithm for maximizing submodular functions still applies and we can strengthen approximation guarantee from Theorem 3.2 to $(1 - e^{-\max(\frac{pK}{|X|}, 1)})$. Theorem 3.2 has interesting implication: if we restrict our problem to the class of instances for which $\frac{p}{|X|}$ is lower-bounded by some constant, then there exists a polynomial time approximation scheme (PTAS) for such a restricted problem. This observation is interesting, because for some real-life problems it is more natural to express the value of the parameter p as the fraction of the size of the universe set X rather than as the absolute value.

Corollary 3.3. *Let $\gamma \in \mathbb{R}$ be a constant. There exists a polynomial time approximation scheme for the maximization variant of the BESTKSUBSET problem with non-negative, non-decreasing, and at-least- $(\gamma|X|)$ -subseparable set function.*

Proof. For each ϵ , $0 < \epsilon < 1$, there exists such a constant c that for $K > c$, $(1 - e^{-\gamma K}) \geq 1 - \epsilon$. For $K > c$ we run the greedy algorithm (and the approximation ratio follows from Theorem 3.2), and for $K \leq c$, we invoke a brute-force algorithm that tries all K -element subsets of X . \square

Finally, we consider the minimization variant of BESTKSUBSET for the case of *at-most- p -subseparable* submodular set functions. In Algorithm 3 we present a randomized algorithm for the problem: the algorithm performs several independent runs. Each run, in Algorithm 3 described by the SingleRun procedure, builds the solution by selecting random elements in K consecutive steps. In each step, an element x is selected with the probability proportional to the marginal increase of the value of the set function caused by adding x to the partial solution. Theorem 3.4 below shows that if we repeat such a procedure a sufficient number of times, we are very likely to find a solution with the required approximation ratio.

Theorem 3.4. *For each non-negative, non-decreasing, submodular, at-most- p -subseparable set function $v : 2^X \rightarrow \mathbb{R}$ and for each $0 \leq \beta < 1$, Algorithm 3 outputs a β -approximate solution*

for the minimization variant of BESTKSUBSET, with probability $(1 - \epsilon)$. The time complexity of the algorithm is $\text{poly}(n, m) \cdot \lceil -\ln \epsilon / (\frac{\beta-1}{p\beta})^K \rceil$.

Proof. Let I be an instance of the BESTKSUBSET problem with $v : 2^X \rightarrow \mathbb{R}$ being a non-negative, submodular, at-most- p -subseparable function. Let $\beta, \beta > 1$, and $\epsilon, 0 < \epsilon < 1$ be the parameters of Algorithm 3. Let S^* be some optimal solution for I .

Let us consider a single call to `SingleRun` from the “for” loop within the function `Main`. Let p_s denote the probability that such a single invocation of the function `SingleRun` returns a β -approximate solution. We will prove the lower-bound of $(\frac{\beta-1}{p\beta})^K$ for the value of p_s . Let Ev denote the event that during such an invocation, at the beginning of each iteration of the “for” loop within the function `SingleRun`, it holds that:

$$v(X) - v(S) > \beta(v(X) - v(S^*)). \quad (4)$$

Note that if the complementary event, denoted \overline{Ev} , occurs, then `SingleRun` definitely returns a β -approximate solution. Condition in Inequality 4 can be reformulated as follows:

$$\frac{v(S^*) - v(S)}{v(X) - v(S)} > \frac{\beta - 1}{\beta}. \quad (5)$$

Now, let us consider a single iteration of the “for” loop within the function `SingleRun`. Let S be the value of the partial solution at the beginning of this iteration and let p_{hit} denote the probability that in this iteration the element from S^* is added to the partial solution (thus, using notation from Algorithm 3, p_{hit} is the probability that $x_r \in S^*$). Let us assess the conditional probability $p_{hit|Ev}$:

$$\begin{aligned} p_{hit|Ev} &= \frac{\sum_{x \in S^*} (v(S \cup \{x\}) - v(S))}{\sum_{x \in X} (v(S \cup \{x\}) - v(S))} \\ &\geq \frac{(v(S \cup \{x_1\}) - v(S)) + (v(S \cup \{x_1, x_2\}) - v(S \cup \{x_1\})) + \dots}{\sum_{x \in X} (v(S \cup \{x\}) - v(S))} && \text{submodularity} \\ &= \frac{v(S \cup S^*) - v(S)}{\sum_{x \in X} (v(S \cup \{x\}) - v(S))} \\ &\geq \frac{v(S^*) - v(S)}{\sum_{x \in X} (v(S \cup \{x\}) - v(S))} && \text{non-decreasing} \\ &\geq \frac{v(S^*) - v(S)}{p(v(X) - v(S))} && \text{at-most-}p\text{-subseparability} \\ &\geq \frac{\beta - 1}{p\beta}. && \text{Equation 5} \end{aligned}$$

Let p_{opt} denote the probability that the function `SingleRun` returns S^* , an optimal solution. We have that:

$$p_{\text{opt}|Ev} \geq (p_{\text{hit}|Ev})^K \geq \left(\frac{\beta-1}{p\beta}\right)^K.$$

Altogether, combining all the above findings, we know that the probability that `SingleRun` returns a β -approximate solution is at least:

$$p_s \geq P(\overline{Ev}) + P(Ev)p_{\text{opt}|Ev} \geq p_{\text{opt}|Ev} \geq \left(\frac{\beta-1}{p\beta}\right)^K. \quad (6)$$

The estimation in Inequality 6 bases on the fact that either the event \overline{Ev} happens, and in such case `SingleRun` definitely returns a β -approximate solution, or \overline{Ev} does not occur (thus, Ev happens), and in such case we can lower-bound the probability of finding a β -approximate solution by the probability of finding an optimal one.

To conclude, we use the standard argument that if we make $x = \lceil \frac{-\ln \epsilon}{p_s} \rceil$ independent calls to `SingleRun`, then the best output from these calls is a β -approximate solution with probability at least equal to:

$$1 - (1 - p_s)^x \geq 1 - e^{-\ln \epsilon} = 1 - \epsilon.$$

This completes the proof. \square

Interestingly, we can slightly modify the proof of Theorem 3.4 so that it would apply with the more general parameter $\frac{\sum_{x \in X} v(\{x\})}{v(X)}$. On the other hand, for this parameter we give weaker approximation guarantees, by approximating the minimization-or-maximization instead of the minimization variant of the problem.

Theorem 3.5. *For each non-negative, non-decreasing and submodular set function $v : 2^X \rightarrow \mathbb{R}$ there exists an FPT approximation scheme for the minimization-or-maximization variant of BESTK-SUBSET problem with the parameter $(K, \frac{\sum_{x \in X} v(\{x\})}{v(X)})$.*

Proof. Let us fix $\beta, \beta > 1$, the required approximation ratio. Let $p = \frac{\beta}{\beta-1} \cdot \frac{\sum_{x \in X} v(\{x\})}{v(X)}$. We will show that Algorithm 3 with such value of the parameter p (this parameter is used to determine the number of iterations of the algorithm) is a β -approximation algorithm for the minimization-or-maximization variant of the problem. We repeat the reasoning from the proof of Theorem 3.4, with the following small modification. In the proof of Theorem 3.4 we defined Ev to denote the event that during a single invocation of the `SingleRun` function from Algorithm 3, at the beginning of each iteration of the “for” loop, it holds that: $v(X) - v(S) > \beta(v(X) - v(S^*))$. In this proof we modify this definition saying that Ev denotes the event when at the beginning of each iteration of the “for” loop within the function `SingleRun`, the following *two* conditions hold:

$$v(X) - v(S) > \beta(v(X) - v(S^*)),$$

$$v(S) < \frac{1}{\beta} v(S^*).$$

Naturally, if the complementary event occurs, then `SingleRun` definitely returns a β -approximate solution for the minimization-or-maximization variant of the problem. In the proof of Theorem 3.4, we used at-most- p -subseparability in the part that assumes that the event Ev happened, to show that:

$$\sum_{x \in X} \left(v(S \cup \{x\}) - v(S) \right) \leq p \left(v(X) - v(S) \right) \quad (7)$$

Here, we show that Inequality 7 also holds if we assume that the event Ev (using our redefinition of Ev) happened:

$$\begin{aligned} \sum_{x \in X} \left(v(S \cup \{x\}) - v(S) \right) &\leq \sum_{x \in X} \left(v(\{x\}) - v(\emptyset) \right) \leq \sum_{x \in X} v(\{x\}) \\ &= p \cdot \frac{\beta - 1}{\beta} \cdot v(X) = p \cdot v(X) - p \cdot \frac{v(X)}{\beta} \\ &\leq p \cdot v(X) - p \cdot v(S). \end{aligned}$$

With these modifications the proof of Theorem 3.4 can be used in this case. \square

Algorithm 3 can be applied to yet another variant of the problem. Let `BESTSUBSET` be defined similarly to `BESTKSUBSET`, with the following difference. In `BESTSUBSET` we are not putting any constraints on the size of the solution, but we rather look for the smallest possible set S such that $v(S) = v(X)$. Interestingly, Algorithm 3 can be used to find *exact* solutions to `BESTSUBSET` for non-negative, non-decreasing, submodular, at-most- p -subseparable set functions, and it will run in FPT time for the parameter (K, p) .

Theorem 3.6. *For each non-negative, non-decreasing, submodular, at-most- p -subseparable set function $v : 2^X \rightarrow \mathbb{R}$, the algorithm that runs Algorithm 3 for consecutive values of the parameter K until it finds a solution S , such that $v(S) = v(X)$, is a randomized FPT exact algorithm for the `BESTSUBSET` problem for the parameter (K, p) .*

Proof. Let S^* be an optimal solution for the problem. Let us consider a single iteration of the “for” loop within the function `SingleRun`, and let S be the value of the partial solution at the beginning of this iteration. We define p_{hit} as the probability that in iteration, an element from S^* is added to the partial solution. We can use a very similar estimation as in the proof of Theorem 3.4:

$$\begin{aligned} p_{hit} &= \frac{\sum_{x \in S^*} \left(v(S \cup \{x\}) - v(S) \right)}{\sum_{x \in X} \left(v(S \cup \{x\}) - v(S) \right)} \\ &\geq \frac{\left(v(S \cup \{x_1\}) - v(S) \right) + \left(v(S \cup \{x_1, x_2\}) - v(S \cup \{x_1\}) \right) + \dots}{\sum_{x \in X} \left(v(S \cup \{x\}) - v(S) \right)} \quad \text{submodularity} \end{aligned}$$

$$\begin{aligned}
&= \frac{v(S \cup S^*) - v(S)}{\sum_{x \in X} (v(S \cup \{x\}) - v(S))} \\
&\geq \frac{v(S^*) - v(S)}{\sum_{x \in X} (v(S \cup \{x\}) - v(S))} && \text{non-decreasing} \\
&\geq \frac{v(S^*) - v(S)}{p(v(X) - v(S))} && \text{at-most-}p\text{-subseparability} \\
&\geq \frac{1}{p}. && \text{since } v(S^*) = v(X)
\end{aligned}$$

Thus, we can lower bound the probability that S^* will be found by a single run of the function `SingleRun`, by $\frac{1}{p^K}$. If we invoke `SingleRun` a sufficient number of times, then the probability of not finding an optimal solution can be decimated. \square

4 Applications of the Algorithms

In this section we show that the assumption about p -separability of submodular set functions is plausible. We provide several examples of known computational problems that can be expressed as maximization of p -separable, submodular functions. Consequently, we show that the algorithms from Section 3 are applicable to these problems.

4.1 Item Selection in Multi-Agent Systems

Let $N = \{1, 2, \dots, n\}$ be the set of agents and let $C = \{a_1, a_2, \dots, a_m\}$ be the set of *items*. Each agent $i \in N$ is endowed with a *utility function* $u_i : C \rightarrow \mathbb{R}$ that measures how much i desires each of the items. Our goal is to select K items, called *winners*, that in some sense would make the agents most satisfied. Naturally, there are various ways to define satisfaction of the agents. In this section we follow a recent model of Skowron et al. [35] which describes a broad variant of this problem, parameterized by the ordered weighted average (OWA) vectors.

An OWA vector α is a vector of K elements, $\alpha = \langle \alpha_1, \dots, \alpha_K \rangle$. Given an OWA vector α , for each agent i and for each set of K items S , we define $u_i(S)$, the satisfaction of i from S , in the following way. Let u_1, u_2, \dots, u_K be the utilities from $\{u_i(x) : x \in S\}$, sorted in the descending order; then $u_i(S) = \sum_{j=1}^K \alpha_j u_j$. The satisfaction of all agents from S is defined as the sum of satisfactions of all the individuals from S .

This model captures various natural problems, from winner determination in multiwinner election systems, through recommendation systems, to location problems. For instance the problem of selecting K items under the OWA vector $\alpha = \langle 1, 0, \dots, 0 \rangle$ boils down to the problem of winner determination under Chamberlin and Courant rule [6], or to the facility location problem [16, 33, 7]. The problem for $\alpha = \langle 1, 1/2, \dots, 1/K \rangle$ is equivalent to winner determination in the Proportional

Approval Voting (PAV) system [18]. For more examples of applications of this general model we refer the reader to the original work of Skowron et al. [35].

We say that the agents have k -approval utilities if each agent assigns utility equal to 1 to exactly k items, and utility equal to 0 to the remaining ones. Such k -approval utilities are very popular in the context of social choice, in particular in case of multi-winner election rules [18, 4, 1, 34, 2].

Below, we show that Algorithms 1 and 3 are applicable to the problem of selecting K items in the OWA-based framework with k -approval utilities.

Lemma 4.1. *For a non-decreasing OWA and for k -approval utilities, the problem of selecting K best items can be expressed as the maximization of a nonnegative, nondecreasing submodular function which is (i) k -superseparable, and (ii) at-most- k -subseparable.*

Proof. We start from defining an appropriate set function v . It is easy to extend the definition of the utility of an agent i from a set of items S , to cover sets with a different number of elements than K : $u_i(S) \sum_{j=1}^{\min(K, |S|)} \alpha_j u_j$. For each set $S \subseteq C$, we define $v(S)$ as the total satisfaction of the agents from S : $v(S) = \sum_{i \in N} u_i(S)$.

Skowron et al. [34] showed that such defined function is submodular. We will show that it is also k -superseparable and at-most- k -subseparable. It is easy to see that the sum of p -superseparable set functions is also p -superseparable. The same argument applies to at-most- p -subseparability. Thus, it suffices to show that our hypothesis is true for an arbitrary k -approval utility function u_i .

We fix $S \subseteq C$: let ℓ denote the number of elements in S which are approved of by i . Since, the utilities of the agents are k -approval, there are only $(k - \ell)$ elements in $C \setminus S$ for which $u_i(\{x\}) > 0$. For each such an element x , we have $(v(S \cup \{x\}) - v(S)) = \alpha_{\ell+1}$. Thus:

$$\sum_{x \in X} (u_i(S \cup \{x\}) - u_i(S)) = (k - \ell)\alpha_{\ell+1}.$$

If $\ell \geq 1$, then:

$$\sum_{x \in X} (u_i(S \cup \{x\}) - u_i(S)) = (k - \ell)\alpha_{\ell+1} \geq 0 = k\alpha_1 - k\alpha_1 \geq \sum_{x \in X} u_i(\{x\}) - ku_i(S).$$

If $\ell \geq 0$, then:

$$\sum_{x \in X} (u_i(S \cup \{x\}) - u_i(S)) = k\alpha_1 = \sum_{x \in X} u_i(\{x\}) \geq \sum_{x \in X} u_i(\{x\}) - ku_i(S).$$

Which shows that u_i is k -superseparable, and thus that v is k -superseparable. Further,

$$\begin{aligned} \sum_{x \in X} (u_i(S \cup \{x\}) - u_i(S)) &= (k - \ell)\alpha_{\ell+1} \leq (k - \ell) \left(\sum_j \alpha_j - \sum_{j \leq \ell} \alpha_j \right) \\ &= (k - \ell)(u_i(X) - u_i(S)) \leq k(u_i(X) - u_i(S)). \end{aligned}$$

Which shows that u_i is at-most- k -subseparable. This completes the proof. \square

As the corollary we get that Algorithms 1 and 3 are applicable to the problem.

Corollary 4.2. *There exists an FPT approximation scheme for the maximization and minimization variants of the problem of selecting K items with k -approval utilities for the parameter (K, k) .*

There are numerous consequences of Corollary 4.2. First, Algorithms 1 and 3 are applicable to the facility location problem [16]. Further, since the item selection problem is a model for multi-winner election systems (with candidates corresponding to items and voters to agents), Algorithms 1 and 3 are applicable to the problem of finding winners under Chamberlin and Courant [6] and Proportional Approval Voting [18] election systems. In all these cases the assumption that the number of approved items is small is realistic. For instance, in case of elections, in some countries, the voting procedure imposes constraints on how many candidates a voter can approve of (for instance, in the Polish parliamentary elections these are only three candidates). Our results can be also extended to cover geometric utilities, that is the case where the set of utilities of each agent has the form $\{d^{m-1}, d^{m-2}, \dots, 1\}$, for some $d > 1$.

4.2 Matching and Assignment Problems

The algorithm for p -superseparable set functions, i.e., Algorithm 1, is also applicable to variants of assignment problems in bipartite graphs. We provide an example by showing how to apply our results to the WEIGHTED-B-MATCHING problem, which is similar in spirit to the item selection problem from the previous subsection. Here, however, we introduce additional capacity constraints, which say that items cannot be shared among too many agents.

In the WEIGHTED-B-MATCHING problem we are given a set of vertices $X \cup Y$, a set of edges E (there are no edges neither between the vertices from X nor between the vertices from Y), a weight function $w : E \rightarrow \mathbb{R}$, and a capacity function $c : X \rightarrow \mathbb{Z}$. The goal is to find a subset of edges with the maximal total weight, such that each vertex $x \in X$ belongs to at most $c(x)$ of the selected edges, each vertex $y \in Y$ belongs to at most one of the selected edges, and altogether there are at most K vertices from X which belong to some of the selected edges.

Let us explain the relation between the WEIGHTED-B-MATCHING problem and the item selection problem from the previous subsection. We observe that vertices from X can be thought of as items and the vertices from Y can be identified with agents. A weight w of an edge (x, y) , $x \in X$ and $y \in Y$, corresponds to the utility that the agent y assigns to the item x .

Lemma 4.3. *The WEIGHTED-B-MATCHING problem with the degree of vertices from Y bounded by p can be expressed as the maximization of a nonnegative, nondecreasing submodular function which is p -superseparable.*

Proof. For each set $S \subseteq X$ we define $v(S)$ as the maximum weight of a matching for the graph induced by the set of vertices $S \cup Y$. Such defined v is nonnegative and submodular [36].

First, we show that v is p -superseparable. For each S , let \mathcal{M} denote some optimal matching for the graph induced by the set of vertices $S \cup Y$. For each S , and for each $x \in X$, $(v(S) + v(\{x\}) - v(S \cup \{x\}))$ is no greater than the weight of such edges $(x', y') \in \mathcal{M}$ that there exists an edge between x and y' . This holds because we can construct a feasible matching for the graph induced by $S \cup \{x\} \cup Y$ in the following way. Let us denote the matchings for the graphs induced by $S \cup Y$ and by $\{x\} \cup Y$, as \mathcal{M}_1 and \mathcal{M}_2 respectively. If we merge \mathcal{M}_1 and \mathcal{M}_2 and drop each edge (x', y')

from \mathcal{M}_1 that connects a vertex $y' \in Y$ which is already connected in \mathcal{M}_2 (i.e., each edge (x', y') from \mathcal{M}_1 such that there exists an edge (x, y') in \mathcal{M}_2), we get a feasible matching for the graph induced by $S \cup \{x\} \cup Y$. Consequently, $\sum_{x \in X} (v(S) + v(\{x\}) - v(S \cup \{x\}))$ is no greater than the weight of \mathcal{M}_1 times the bound on the degree of vertices from Y . That is:

$$\sum_{x \in X} (v(S) + v(\{x\}) - v(S \cup \{x\})) \leq p \cdot v(S),$$

which is equivalent to the condition for p -superseparability. This proves the thesis. \square

Corollary 4.4. *There exists an FPT approximation scheme for the maximization variant of the WEIGHTED-B-MATCHING for the parameter p , the bound on the degree of vertices from Y .*

Since finding winners under the Monroe election system [28] is computationally equivalent to solving WEIGHTED-B-MATCHING with specific weights and specific capacities [36], Corollary 4.4 implies that there exists an FPT approximation scheme for winner determination under Monroe election system with k -approval utilities, for the parameter (k, K) .

4.3 The MAXWEIGHTCOVER Problem

In this subsection we show that all our algorithms are applicable to MAXWEIGHTCOVER, a generalized variant of the MAXCOVER problem.

In the MAXWEIGHTCOVER problem, we are given a universe set $N = \{e_1, e_2, \dots, e_n\}$ of n elements and a collection $X = \{S_1, \dots, S_m\}$ of m subsets of N . Each element e_i has its weight w_i . The goal is to find a subcollection \mathcal{C} of X of size at most K that maximizes the total weight of covered elements: $\sum_{i: i \in S \text{ for some } S \in \mathcal{C}} w_i$.

A *frequency* of an element e_i is the number of sets that contain e_i . Frequency of elements is a natural parameter considered in the context of approximability of covering problems [39]. To the best of our knowledge, for polynomial-time algorithms, there exists no better guarantee for the MAXCOVER problem with bounded frequencies of elements than $(1 - 1/e)$. This is specifically interesting, since such an approximation algorithm exists for the very similar problem SETCOVER [39].

Lemma 4.5. *The MAXWEIGHTCOVER problem with the frequency of elements upper-bounded by p can be expressed as the maximization of a nonnegative, nondecreasing submodular function which is (i) p -superseparable, and (ii) at-most- p -subseparable. The MAXWEIGHTCOVER problem with the frequency of elements lower-bounded by p can be expressed as the maximization of a nonnegative, nondecreasing submodular function which is at-least- p -subseparable.*

Proof. For each set $\mathcal{C} \subseteq X$ we define $v(\mathcal{C})$ as the total weight of elements covered by the sets from \mathcal{C} . Such defined v is nonnegative and submodular.

Similarly, as in the proof of Lemma 4.1 we observe that the weighted sum of p -superseparable set functions is also p -superseparable, and that the same argument applies to at-most- p -subseparability and at-least- p -subseparability. Thus, it is sufficient to consider a function u_i which returns 1 for collections of sets that cover e_i , and 0 for the remaining ones. It is easy to see that if

the frequency of the elements is bounded by p , then $\sum_{S \in X} u_i(\{S\})$, which is the number of sets that cover e_i , is also bounded by p .

Let us fix a collection of sets $\mathcal{C} \subseteq X$ and let us consider two cases. If e_i is covered by \mathcal{C} , then $\sum_{S \in X} (u_i(\mathcal{C} \cup \{S\}) - u_i(\mathcal{C}))$ is equal to 0. But, in such case $pu_i(\mathcal{C}) = p$ and the condition for p -superseparability holds. Naturally, $u_i(X) = 1$, thus the conditions for p -subseparability and at-least- p -subseparability also hold.

If e_i is not covered by \mathcal{C} , then $\sum_{S \in X} (u_i(\mathcal{C} \cup \{S\}) - u_i(\mathcal{C}))$ is equal to the number of sets that cover e_i , thus to $\sum_{S \in X} u_i(\{S\})$. This means that the condition for p -superseparability holds. If the frequency of the elements is upper-bounded by p , then $\sum_{S \in X} (u_i(\mathcal{C} \cup \{S\}) - u_i(\mathcal{C}))$ is upper bounded by p , and since $u_i(\mathcal{C}) = 0$, the condition for at-most- p -subseparability holds. If the frequency of the elements is upper-bounded by p , then $\sum_{S \in X} (u_i(\mathcal{C} \cup \{S\}) - u_i(\mathcal{C}))$ is lower bounded by p , and the condition for at-least- p -subseparability holds. This proves the thesis. \square

Corollary 4.6. *There exists an FPT approximation scheme for the maximization and minimization variant of the MAXWEIGHTCOVER for the parameter (K, p) , where p is the upper-bound on the frequency of the elements.*

As an implication from Corollary 3.3, we also get that:

Corollary 4.7. *Let γ be a constant and let γ -MAXWEIGHTCOVER be the variant of the MAXWEIGHTCOVER problem with additional assumption that each element is covered by at least γ fraction of the sets from X . There exists a PTAS for γ -MAXWEIGHTCOVER.*

Corollaries 4.6 and 4.7 extend the recent results for the MAXCOVER problem [34]. Interestingly, Theorem 3.5 says that there exists a randomized FPT approximation scheme for the minimization-or-maximization variant of the MAXWEIGHTCOVER problem, for the parameter (K, p_{av}) , where p_{av} is an average frequency of an element.

5 Conclusions

In this paper we have considered approximation algorithms for the problem of maximizing of submodular set functions. Since it is known that the greedy algorithm achieves approximation ratio of $(1 - 1/e)$ and that no polynomial-time algorithm can approximate the problem better, we have initiated the study of algorithms for the problem that run in super-polynomial time. In our study we have followed the approach of parameterized complexity theory. We have identified three new properties of set functions, called p -separability properties, and we have considered the parameter p from the definition of p -separability. For p combined with K , the size of the solution, we have shown that the problem can be arbitrarily well approximated by algorithms that run in FPT time. We have justified our choice of the parameters by providing numerous examples of computational problems which can be expressed as maximization of submodular p -separable set functions, and by arguing that in many real-life problems the value of p is small.

We have exposed the difference between approximating the maximization and minimization variants of the problem and we have provided a new weaker definition of approximability: approximation of minimization-or-maximization variant of the problem. We have shown that it is possible

to approximate our problem arbitrarily well using algorithms that run in FPT time for the more general parameter $(K, \sum_{x \in X} v(\{x\})/v(X))$.

There are many natural ways in which this research can be extended. We believe that one of the promising approaches is to consider the problem with additional constraints, such as knapsack constraints or matroid constraints.

References

- [1] H. Aziz, M. Brill, V. Conitzer, E. Elkind, R. Freeman, and T. Walsh. Justified representation in approval-based committee voting. In *Proceedings of the 29th Conference on Artificial Intelligence (AAAI-2015)*, pages 784–790, 2015.
- [2] H. Aziz, S. Gaspers, J. Gudmundsson, S. Mackenzie, N. Mattei, and T. Walsh. Computational aspects of multi-winner approval voting. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems (AAMAS-2015)*, pages 107–115, 2015.
- [3] F. Bach. Structured sparsity-inducing norms through submodular functions. In *Proceedings of Advances in Neural Information Processing Systems 23 (NIPS-2010)*, pages 118–126, 2010.
- [4] D. Baumeister, G. Erdélyi, E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. Computational aspects of approval voting. In J. Laslier and R. Sanver, editors, *Handbook of Approval Voting*, pages 199–251. Springer, 2010.
- [5] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák. Maximizing a submodular set function subject to a matroid constraint (extended abstract). In *Proceedings of the 12th International Conference on Integer Programming and Combinatorial Optimization (IPCO-2007)*, pages 182–196, 2007.
- [6] B. Chamberlin and P. Courant. Representative deliberations and representative decisions: Proportional representation and the borda rule. *American Political Science Review*, 77(3):718–733, 1983.
- [7] F. Chudak and D. P. Williamson. Improved approximation algorithms for capacitated facility location problems. *Mathematical Programming*, 102(2):207–222, Mar. 2005.
- [8] V. Chvatal. A Greedy Heuristic for the Set-Covering Problem. *Mathematics of Operations Research*, 4(3):233–235, 1979.
- [9] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer, 2015.
- [10] A. Das and D. Kempe. Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection. In L. Getoor and T. Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1057–1064, New York, NY, USA, 2011.

- [11] R. Downey and M. Fellows. *Parameterized Complexity*. Springer-Verlag, 1999.
- [12] U. Feige. A threshold of $\ln n$ for approximating set cover. *J. ACM*, 45(4):634–652, 1998.
- [13] U. Feige, V. S. Mirrokni, and J. Vondrák. Maximizing non-monotone submodular functions. *SIAM Journal on Computing*, 40(4):1133–1153, 2011.
- [14] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer-Verlag, 2006.
- [15] S. Iwata, L. Fleischer, and S. Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *J. ACM*, 48(4):761–777, July 2001.
- [16] K. Jain and V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and lagrangian relaxation. *J. ACM*, 48(2):274–296, Mar. 2001.
- [17] S. Jegelka and J. A. Bilmes. Submodularity beyond submodular energies: Coupling edges in graph cuts. In *Proceedings of the 24th IEEE Conference on Computer Vision and Pattern Recognition (CVPR-2011)*, pages 1897–1904, 2011.
- [18] D. M. Kilgour. Approval balloting for multi-winner elections. In J. Laslier and R. Sanver, editors, *Handbook on Approval Voting*, pages 105–124. Springer, 2010.
- [19] G. Kim, E. P. Xing, L. Fei-Fei, and T. Kanade. Distributed cosegmentation via submodular optimization on anisotropic diffusion. In *Proceedings of IEEE International Conference on Computer Vision (ICCV-2011)*, pages 169–176, 2011.
- [20] A. Krause and D. Golovin. Submodular function maximization. Technical report, 2012.
- [21] A. Krause and C. Guestrin. Near-optimal value of information in graphical models. In *Proceedings of Conference on Uncertainty in Artificial Intelligence (UAI-2005)*, July 2005.
- [22] A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9:235–284, June 2008.
- [23] R. Kumar, B. Moseley, S. Vassilvitskii, and A. Vattani. Fast greedy algorithms in mapreduce and streaming. In *Proceedings of the Twenty-fifth Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA-2013)*, pages 1–10, 2013.
- [24] J. Lee, V. S. Mirrokni, V. Nagarajan, and M. Sviridenko. Non-monotone submodular maximization under matroid and knapsack constraints. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing (STOC-2009)*, pages 323–332, 2009.
- [25] H. Lin and J. Bilmes. Multi-document summarization via budgeted maximization of submodular functions. In *Proceedings of Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HTL-2010)*, pages 912–920, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

- [26] H. Lin and J. A. Bilmes. How to select a good training-data subset for transcription: submodular active selection for sequences. In *Proceedings of 10th Annual Conference of the International Speech Communication Association (INTERSPEECH-2009)*, pages 2859–2862, 2009.
- [27] T. Lu and C. Boutilier. Budgeted social choice: From consensus to personalized decision making. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI-2011)*, pages 280–286, 2011.
- [28] B. Monroe. Fully proportional representation. *American Political Science Review*, 89(4):925–940, 1995.
- [29] M. Narasimhan, J. Nebojsa, and B. A. Jeff. Q-clustering. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Proceedings of Advances in Neural Information Processing Systems 18 (NIPS-2010)*, pages 979–986. MIT Press, 2006.
- [30] G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, 14(1):265–294, December 1978.
- [31] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.
- [32] s. Jegelka, F. Bach, and S. Sra. Reflection methods for user-friendly submodular optimization. In C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, editors, *Proceedings of Advances in Neural Information Processing Systems 26 (NIPS-2013)*, pages 1313–1321. Curran Associates, Inc., 2013.
- [33] D. Shmoys, É. Tardos, and K. Aardal. Approximation algorithms for facility location problems (extended abstract). In *Proceedings of the 29th Symposium on Theory of Computing (STOC-1997)*, pages 265–274, 1997.
- [34] P. Skowron and P. Faliszewski. Fully proportional representation with approval ballots: Approximating the maxcover problem with bounded frequencies in FPT time. In *Proceedings of the 29th Conference on Artificial Intelligence (AAAI-2015)*, pages 2124–2130, 2015.
- [35] P. Skowron, P. Faliszewski, and J. Lang. Finding a collective set of items: From proportional multirepresentation to group recommendation. In *Proceedings of the 29th Conference on Artificial Intelligence (AAAI-2015)*, 2015.
- [36] P. Skowron, P. Faliszewski, and A. Slinko. Achieving fully proportional representation: Approximability result. *Artificial Intelligence*, 222:67–103, 2015.
- [37] M. J. Streeter and D. Golovin. An online algorithm for maximizing submodular functions. In *Proceedings of Advances in Neural Information Processing Systems 21 (NIPS-2008)*, pages 1577–1584, 2008.
- [38] M. Sviridenko. A note on maximizing a submodular set function subject to a knapsack constraint. *Oper. Res. Lett.*, 32(1):41–43, 2004.

- [39] V. V. Vazirani. *Approximation algorithms*. Springer-Verlag New York, Inc., New York, NY, USA, 2001.
- [40] J. Vondrák, C. Chekuri, and R. Zenklusen. Submodular function maximization via the multi-linear relaxation and contention resolution schemes. In *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing (STOC-2011)*, pages 783–792, 2011.
- [41] Y. Yue and C. Guestrin. Linear submodular bandits and their application to diversified retrieval. In *Proceedings of Advances in Neural Information Processing Systems 24 (NIPS-2011)*, pages 2483–2491, 2011.